20th International Congress on Modelling and Simulation, Adelaide, Australia, 1–6 December 2013
www.mssanz.org.au/modsim2013

Closing the Loop with the Close Action Environment

L. Finlay [a]

[a] *DSTO, West Ave, Edinburgh, South Australia 5111*
Email: luke.finlay@dsto.defence.gov.au

Abstract: In this paper, I present the Close Action Environment (CAEn) Closed Loop (CAEnCL) wargame simulation that extends CAEn by adding a closed loop layer. An example of its use is presented to demonstrate the reactive branching ability of CAEnCL although the example is not based on a realistic military scenario. The CAEn simulated wargame is typically operated by human players in order to capture a real-time enacted Scheme of Manoeuvre (SoM). In military terms, the SoM defines a plan but a human-in-the-loop CAEn wargame only captures one enactment of the planned SoM. The captured SoM is usually replicated multiple times with different random seeds (which randomise many stochastic components of CAEn's models) to facilitate statistical comparisons between warfighting options under consideration. This approach introduces the potential for non-realistic actions by entities which are blindly following the captured SoM in the replicated simulation, and this inability to make reactive SoM changes limits the confidence that can be placed in the results from CAEn studies. For example, in the captured SoM, a tank may have been destroyed, causing the player to modify their SoM for the remainder of the wargame. However, later when the SoM is replicated, the tank was not destroyed in most replications but the decision to modify their strategy remains.

CAEn Closed Loop (CAEnCL) was developed to address this problem and also enhance the tools currently available to scenario developers and experimenters. CAEnCL connects to a CAEn server and mimics a human player reacting to the current situation by executing pre-defined orders once their trigger requirements are met. Thus, CAEnCL extends CAEn and gives entities the ability to deviate from a generic SoM in response to situational changes.

The scenario developer is provided with the tools to build a complex scenario which includes multiple branches via a Graphical User Interface (GUI). An intricate communications model is also provided, capable of simulating many concurrent communication architectures, facilitating the realistic exchange of information between battlefield entities and resulting in more informed decision making.

A common use of CAEn involves recording two SoMs under different experimental conditions from two human-in-the-loop wargames and comparing the results. Conducting the experiment in this way introduces human variability. CAEnCL assists the experimenter, increasing the confidence in CAEn results by simulating realistic reactions to the current environment. For CAEn human-in-the-loop experiments, where the experimental variable of interest cannot be changed without recording the SoM again, CAEnCL helps the experimenter remove human variability.

In this paper, CAEn and closed loop wargame simulations are introduced followed by an overview of CAEnCL. Greater detail on how CAEnCL works including ordersets, orders, triggers and the communications model is presented. Finally, an example scenario and an analysis of branching is presented to show the enhanced utility of CAEnCL.

Keywords: *CAEn, closed loop simulation, constructive simulation, wargaming, combat simulation*

1 INTRODUCTION

The Close Action Environment (CAEn), originally developed by the UK and written in C/C++, simulates a small battlefield (1 to 10km^2) using a high resolution (up to 1m^2) 3D environment with variable ground height, trees, shrubs and destructible urban terrain. Multiple sides are modelled, each with different unit and weapon types including vehicles and helicopters. For each unit type, multiple sensors (e.g. human eyes) are modelled for different line of sight situational awareness (SA) calculations. When playing CAEn, each individual soldier is modelled and must be told where to look, how to move and what activity to perform. CAEn has been used extensively by the Defence Science and Technology Organisation (DSTO) (see Coutts et al. (2008, 2010); Pietsch et al. (2010); Finlay (2011)). CAEn employs internationally accepted physical models and, over time, the underlying input data for CAEn's weapon models has been fine tuned. Consequently, there is a reasonable level of confidence in the outcomes from simulated combat.

Most DSTO CAEn studies involve human-in-the-loop wargaming, where humans control many simulated units in real time on all sides (see Shine and Dexter (2007)). In these studies, orders were automatically recorded from human players for many experiment conditions of interest (e.g. one game using rifle a and one game using rifle b). When comparing the outcomes of multiple wargames, a large sample size is required to find statistically significant differences. To increase the sample size, the automatically captured SoM is replicated (normally 200 times) with different random seeds. During replication runs, the orders are executed at the same game time that the human player had executed the order in the original game. Therefore, during replication, the in-game units could behave unrealistically if the game situation had changed significantly. For example, if a player loses a soldier from their section early in the game, the recorded SoM will not contain any orders for that unit after its death. Note that, in CAEn, each individual unit requires orders otherwise they will do nothing. In any of the replications where the unit had survived, they would not have any orders and be underutilised for the remainder of that replication.

In order to overcome the issue of unrealistic behaviour of units, a *closed loop* simulation can be used (also referred to as *constructive simulation* in TTCP (2006)). Closed loop simulations execute a complete scenario without any run-time human input after the scenario designer has chosen the initial input parameters. They differ to replications in that the instructions of the units are defined explicitly as opposed to being recorded from a human player. Internally, a closed loop wargame will have a set of orders that are to be executed when certain conditions are met and allow decision paths to be chosen based on the current outcome of the wargame. CAEn can be used as a closed loop simulation by using its built in triggers, zones and timed specific orders. However, the user interface is difficult to use and there are technical limitations on the types of orders and triggers that can be applied.

DSTO's primary closed loop combat simulation was the Combined Arms and Support Task Force Evaluation Model (CASTFOREM) (TRAC (2005)) which was scheduled to be replaced by Combat XXI (TRAC (2012)). Both CASTFOREM and Combat XXI were developed at the United States Army Training and Doctrine Command Analysis Center (TRAC) in White Sands, New Mexico. CASTFOREM has now been phased out in DSTO and, as of this writing, the Australian version of the Combat XXI data is still in the development phase (although the Combat XXI code-base is mature). In the interim, an effective closed loop wargame simulation for Australia was required.

2 CAEN CLOSED LOOP

DSTO has developed the Australian version of CAEn to a point where the data backing various weapon and sensor models is mature. CAEn Closed Loop (CAEnCL) was developed, in Java, to add closed loop functionality to CAEn without using any of CAEn's existing triggers. The terrain, units, ORBAT and unit locations from a CAEn scenario are employed by CAEnCL. Closed loop functionality is provided by CAEnCL connecting to a CAEn server, via TCP/IP, and imitating a human player using the CAEn GUI. Where CAEn is mentioned in this paper, it refers to all of CAEn's functionality without the new CAEnCL closed loop layer. CAEnCL's enhanced functionality includes:

1. An easy to use and intuitive GUI for scenario creation and modification (see Figure 1).
2. The ability to develop complex branching in the scenario (including cancelling orders, simple Boolean logic and dynamic unit selection).
3. Arbitrary grouping of units in addition to the CAEn unit groupings.

4. Runtime specifiable constants that allow a value to be changed between sets of replication runs. For example, communication delays can be changed.

5. A communications suite that models voice, radio, a digital Battle Management System (BMS) and dynamic Situational Awareness (SA) in addition to CAEn's direct line of sight SA.

6. Unexpectedly, it became clear that a scenario could be constructed as "half closed loop". That is, one or more sides (e.g. the civilians) could be controlled by CAEnCL and the remaining sides could be played as human-in-the-loop. This could be used to construct very consistent and reactionary civilians in a human-in-the-loop wargame although no DSTO study has used this particular feature.

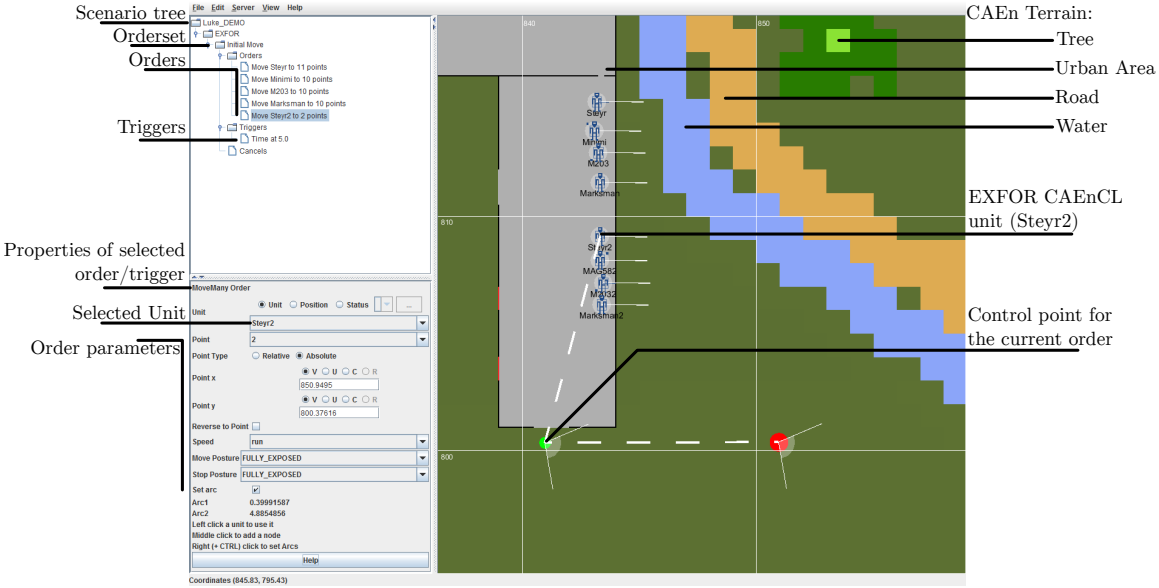

Figure 1. CAEnCL Graphical User Interface.

2.1 CAEnCL Ordersets

When using CAEn, units perform no action or movement unless they receive instructions. By default, in most DSTO developed scenarios, units are set to a state where they will fire the most appropriate available weapon when they identify an enemy but will not move. Movement, posture, surveillance arcs (where a unit looks) and activities need to be entered by humans in real-time when playing a human-in-the-loop CAEn wargame.

In CAEnCL, the scenario designer also needs to enter all of the same information that would normally be entered by human players in order to drive CAEn. To achieve the execution of orders, many *ordersets* are created which contain:

- An orderset definition including name, the number of required triggers and an optional repetition description (which allows the same orderset to be executed many times).
- A set of CAEnCL triggers. Often only one trigger is required but many can be combined with some facility for Boolean logic.
- Optional cancelling triggers which are the same as normal triggers except that, when fired, ensure the orderset is not actioned. To construct a branch, if a particular orderset is fired, the scenario designer will want to cancel the alternate branch.
- A set of orders, all of which are to be executed simultaneously when the containing orderset is actioned. Most of the possible orders map directly to simple CAEn inputs although CAEnCL does have some

more complex orders such as *MoveMany* order which can move a unit and set its surveillance arc while moving.

2.2 CAEnCL Triggers

All triggers can fire with a specified delay to simulate delays in various systems (e.g. people do not generally make instantaneous decisions). Triggers that involve a unit must have a specified unit as a trigger parameter. Triggers that can be used in CAEnCL are:

- **AcquisitionTrigger** Fires when a unit acquires a specified target unit.
- **AcquisitionIsolated** Is normally used for targeting and is similar to the AcquisitionTrigger. It will only fire if friendly units are not close to the target.
- **ActivityTrigger** Fires when a unit starts or finishes a specified activity.
- **CommsMessageTrigger** Fires when a specified message is received by a unit.
- **EmbusTrigger** Fires when a unit is embussed or debussed from a vehicle.
- **HE/Bullet/SmokeTrigger** Fires when a respective weapon effect occurs inside a specified polygon.
- **KillTrigger** Fires when a unit is killed.
- **MoveInTrigger** Fires when a unit is moving inside a specified polygon.
- **NoiseTrigger** Fires when a unit hears a loud noise in a certain direction.
- **OrderSetTrigger** Fires when another orderset is executed.
- **TimeTrigger** Technically a trigger that fires when the game starts. The delay is used to set the fire time.

2.3 CAEnCL Orders

All CAEnCL orders, except for the *End* order require a unit to be selected. CAEnCL orders generally map directly to CAEn orders. Possible orders that can be used in CAEnCL are:

- **ActivityOrder** Set a unit to a particular CAEn activity.
- **AimZone** Set the aim-zone of a unit (used in CAEn for targeted weapons).
- **ClearSA** Clears specified SA of a unit until it is acquired again. This is useful when a unit fires a high explosive weapon at an enemy and assumes the enemy was neutralised.
- **GiveSA** Gives a unit the SA of a target. This can be used to model smoke signatures or mobile phone calls.
- **SetArc** Set the direction a unit is looking.
- **MoveMany** Move a unit along a path of predefined points. The arc can also be set at each point.
- **CommsMessageOrder** Tells a unit to send a message via radio or BMS. This is directly related to the *CommsMessageTrigger* and is generally defined in pairs.
- **StatusUpdateOrder** Only applicable to units with a radio. When this order is executed, the unit says where they are on their radio.
- **End** Ends the game. At least one orderset must contain an *End* order. Since the end can be executed from any number of triggers, complex ending conditions can be implemented easily.

Each of the ordersets is normally triggered by the actions of a "previous" orderset (although ordersets are not defined with ordinality). For example, the scenario designer may want to trigger an action based on when a unit arrives at a specific location x. It is unreasonable to assume that the unit will arrive at location x after a specified amount of time because they may have been delayed (e.g. suppressed). Instead, the scenario designer should place a trigger at the destination of the previous move order, chaining the two ordersets together indirectly. When replicating a CAEn human-in-the-loop wargame, all orders are based on time and strange behaviour can result. For example, a mortar team could be given an order to fire their mortars while inside a two storey building. An example of chained ordersets is given in Section 3 where an example scenario is constructed.

2.4 CAEnCL Communications Model

CAEnCL receives the CAEn SA from each unit's sensors and, if the communications model is enabled, propagates the SA to other units. Therefore, units can be "aware" of another unit without seeing it themselves. On top of the shared SA, ordersets can be fired via the communications models if implemented in the scenario. The communication model simulates voice, radio, a digital Battle Management System (BMS) and the *CommsMessageOrder* (see Section 2.3).

3 EXAMPLE SCENARIO

A simple example CAEnCL scenario is constructed in this section. Both the EXercise FORce (EXFOR) and the OPosing FORce (OPFOR) are defined and basic results from CAEnCL are presented. A graph of the ordersets is shown in Figure 2. Note that this example is a demonstration of the systems branching capability and does not follow any military doctrine.

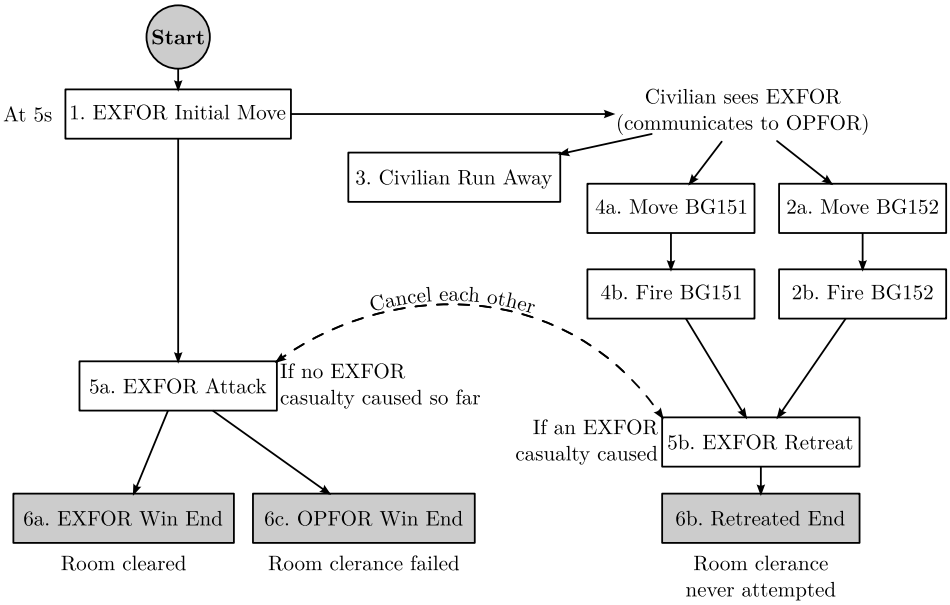

Figure 2. Graph of SoM for all ordersets.

In this scenario, there are three ordersets to model the EXFOR SoM, four ordersets for OPFOR and one for the civilian. In addition, there are three ordersets for different ending conditions. EXFOR's first orderset causes them to move to the outside of the compound of interest. If they do not have any casualties up to this point, they will clear the northern building where two enemy are known to be located. As an example of a branch, if the OPFOR are successful at causing an EXFOR casualty before the clear order is triggered, the remaining EXFOR will retreat. However, once the EXFOR have committed to clearing the building, the retreat branch will no longer be an option.

The single civilian will stand at their initial location and fire a retreat order when they see any EXFOR unit come within 20 metres. Before retreating, they will communicate via voice, each of the locations of any EXFOR unit they see. OPFOR will hear the civilian unit and has two ordersets to move the two southern units outside of their building. Once each unit arrives at their outside point, they will engage the EXFOR with their BG15 (an under-slung grenade launcher attachment). If they do manage to cause an EXFOR casualty, this will cause the EXFOR retreat branch.

Detailed descriptions of every orderset in this scenario is outside the scope of this paper. Instead, the behaviour of one of the OPFOR BG15 units is described in detail. Figure 3 shows the single *Acquisition Trigger* that fires the *BG151 Move* orderset. The observer is the BG151, the target is any EXFOR ("BLUE" in the figure) unit and they must be observed in the red quadrilateral shown to the north of the image. Since the observer is

inside a building, they will never be able to directly observe any EXFOR unit but the civilian standing in the area will inform them what it is seeing (using CAEnCL's communications model). Thus, the BG151 unit will be aware of units in the target location and the trigger will fire.

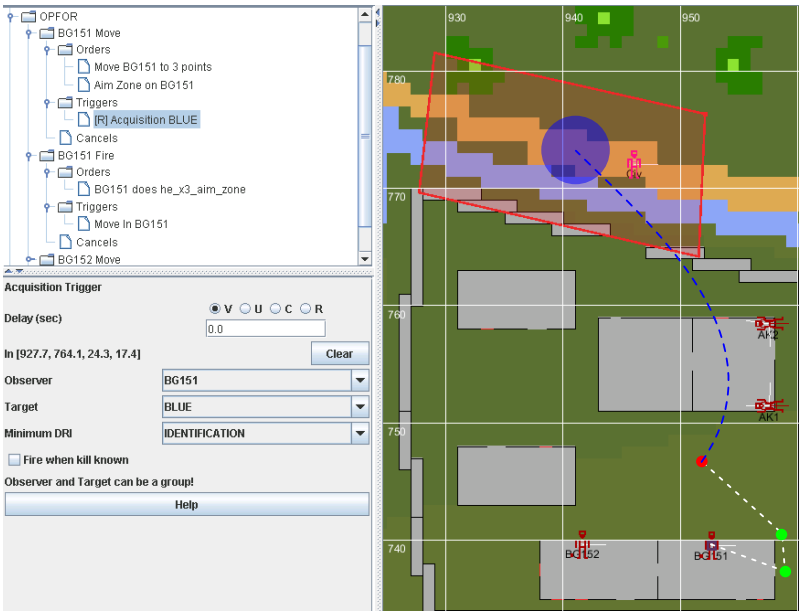

Figure 3. BG151 Move Orderset showing the acquisition trigger. Two orders showing the *move many* and *aim zone* orders are drawn (the orders would not normally be visible in the GUI at this point).

Once the acquisition trigger fires, it stores the location of the observation which is passed to an *Aim Zone* order in the same orderset. That is, the aim zone (to be used for the next orderset) for the BG151 unit is set based on the trigger that fired it. At the same time, a *Move Many* order is executed to move the BG151 unit out into the open.

The *BG151 Fire* orderset has a *Move in* trigger that is fired when the BG151 unit arrives at its destination from the previous orderset. When this orderset is actioned, after a delay, a single action order is executed to fire its BG15 grenade launcher. Delays can be added to any trigger to represent potential real world constraints. For this particular trigger, the delay is drawn from the Poisson distribution with the average $\lambda = 3$ seconds. Note that the aim zone was set in the previous order and is assumed to be set for this order as well (knowing that there are no other ordersets which affect this unit). CAEnCL is only able to pass coordinate information between a trigger and an order if they are in the same orderset. In this scenario, the civilian "knows" OPFOR's plan and an orderset has been created for the civilian to run away as soon as it observes any EXFOR unit nearby.

The details presented above demonstrate the simple manoeuvre of one CAEnCL entity, the BG151. While not overly complicated, it took about two minutes for an experienced CAEnCL user to input. For a more exhaustive list of possible orders and triggers, see Finlay (2013).

3.1 Results

The above example scenario was replicated 200 times with different random seeds. Table 1 shows how many times each of the ordersets (referring back to Figure 2) were executed, sorted in average temporal order. Of interest, the ending conditions show that the EXFOR retreated 34.5% of replications and attacked in 66.5% of replications, demonstrating the primary branching behaviour of this CAEnCL scenario. The two branches could be analysed further by looking at the casualties and other simulation output but that is outside the scope of this paper. Also, the *BG151 Fire* orderset is only executed 98.5% of the time even though the precursory order to move is executed for every replication. The discrepancy of three replications is due to a very long trigger delay, drawn from the Poisson distribution as discussed in Section 3.

Table 1. Example orderset execution count.

Figure 2 reference	Orderset	Count
1	EXFOR Initial Move	200
2a	BG152 Move	200
2b	BG152 Fire	200
3	Civ run	200
4a	BG151 Move	200
4b	BG151 Fire	197
5b	EXFOR Retreat	69
5a	EXFOR Attack	131
6a	EXFOR Win End	126
6b	Retreated End	69
6c	OPFOR Win End	5

If this scenario were to be captured and replicated using a human-in-the-loop CAEn wargame, the branch would not be possible because EXFOR's decision to attack or retreat would be made based on the information in the captured game. In addition, if a BG15 was successful in neutralising an EXFOR unit and the decision was still made to attack, the EXFOR casualty would not be given any orders when the scenario was replicated and therefore be underutilised for the rest of that particular replication. The same situation applies to the OPFOR and civilian units as well. Of course, the captured game will be realistically reactive given that the human player will be making relevant choices but the issue of unrealistic behaviour occurs when the captured SoM is replicated.

4 CONCLUSIONS

This paper has introduced CAEnCL and discussed how it provides a major enhancement of the CAEn simulation. The example scenario demonstrates the benefits of CAEnCL's reactive branching where consistent decisions are made based on the current replication's situation. Additionally, the example scenario demonstrated the usefulness of other CAEnCL features such as the communication model. Consequently, by using CAEnCL, the scenario developer now has the ability to have greater realism in the simulation with greater flexibility in the internal decision points within a scenario.

REFERENCES

Coutts, A., R. Dexter, D. Sanderson, D. Shine, and L. Finlay (2008). Micro Combat Team (MCT) 2012 Experiment 1 - Platoon Operations in a Complex Environment. Technical report, DSTO.

Coutts, A., B. Pietsch, R. Dexter, D. Sanderson, L. Finlay, D. Shine, D. Goodburn, and J. Hansen (2010). Micro Combat Team (MCT) 2012 Experiment 2 - Platoon Operations in a Complex Environment. Technical report, DSTO.

Finlay, L. (2011). *How Changes in Survivability and Lethality Affect Performance in a L400 Setting*. DSTO.

Finlay, L. (2013). *CAEnCL User Guide*. DSTO.

Pietsch, B., L. Finlay, A. Coutts, R. Heyer, D. Shine, and D. Sanderson (2010). Combat Operations in Rural Terrain: Analysis of Micro Combat Team Experiment 2012-3. Technical report, DSTO.

Shine, D. and R. Dexter (2007). *CAEn System Data User Guide v9.2*. DSTO.

TRAC (2005). Combined Arms and Support Task Force Evaluation Model (CASTFOREM) Update: Executive Summary, Department of the Army, USA.

TRAC (2012). COMBATXXI User Guide, Department of the Army, USA.

TTCP (2006). Guide for Understanding and Implementing Defense Experimentation (GUIDEx).